LUCIANA SCHURT

MARIANE GONÇALVES

SANDRO KLOSTERMANN

SISTEMA DOCBOARD:
SISTEMA DE GERENCIAMENTO DE DOCUMENTOS

Trabalho de conclusão de curso de graduação apresentado à disciplina de Projetos do Curso de Tecnologia em Informática, Setor Escola Técnica da Universidade Federal do Paraná.

Orientador: Prof. Dr. Mauro José Belli

CURITIBA

2007

TERMO DE APROVAÇÃO

LUCIANA SCHURT
MARIANE GONÇALVES
SANDRO KLOSTERMANN

SISTEMA DOCBOARD:
SISTEMA DE GERENCIAMENTO DE DOCUMENTOS

Trabalho de Conclusão de Curso aprovado como requisito para obtenção da graduação no Curso de Tecnólogo em Informática, Setor Escola Técnica da Universidade Federal do Paraná, pela seguinte banca examinadora:

Orientador: Prof. Dr. Mauro José Belli
Departamento de Tecnologia, UFPR

Prof. Dieval Guizelini
Departamento de Tecnologia, UFPR

Profª Jeroniza Nunes Marchaukowski
Departamento de Tecnologia, UFPR

Curitiba, 25 de junho de 2007

AGRADECIMENTOS

A todos que durante o período de realização de nosso Trabalho de Conclusão de Curso tiveram paciência com nossas ausências, aos que nos deram todo o suporte e apoio de que precisávamos e que sempre acreditaram em nosso potencial e não nos deixaram desanimar.

Em especial aos nossos companheiros e pais que nos apoiaram e suportaram nossas ausências; ao orientador e ao cliente, que acreditaram em nosso potencial e sanaram muitas das nossas dúvidas e aos nossos chefes que possibilitaram a realização de muitas atividades do projeto durante o horário de trabalho.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

APF – Análise de Pontos por Função

CENADEM - Centro Nacional de Desenvolvimento do Gerenciamento da Informação

CMM – Capability Maturity Model

CMMI – Capability Maturity Model Integration

DER – Diagrama entidade-relacionamento

GED – Gerenciamento eletrônico de documentos

HTML – HyperText Markup Language

IDE – Integrated Development Environment

ISO – International Standards Institute

PGP – Plano de Gerenciamento de Projetos

RSM – Rational Software Modeler

RUP – Rational Unified Process

SQL - Structured Query Language

UML – Unified Modeling Language

WBS – Work Breakdown Structure

RESUMO

Este documento apresenta todo o processo de desenvolvimento de software aplicado ao Projeto de Conclusão de Curso, direcionado a atender o cliente chamado Stefanini IT Solutions.

Foi feita uma análise sobre a necessidade do cliente de um sistema que suprisse as deficiências da empresa com relação ao armazenamento e gerenciamento de documentos editáveis, automatizando alguns processos manuais referentes a fluxos de trabalho. Não existia qualquer controle sobre os documentos gerados nos projetos, que são os produtos gerados pela empresa, havendo riscos de perdas ou alterações na sua documentação. Atualmente a segurança dos documentos é um procedimento vital para uma empresa de sucesso.

Como foco principal, foi abordada toda a metodologia empregada durante o desenvolvimento do projeto, desde a aquisição do cliente, elaboração de um plano de projeto, o entendimento dos objetivos do sistema a ser desenvolvido, até as etapas subseqüentes de análise, implementação e documentação.

Para o desenvolvimento da ferramenta foi utilizada a linguagem C#, com a IDE Visual Studio 2005, rodando sobre a plataforma Windows 2000 ou superior e banco de dados SQL Server 2000.

Por fim é apresentado o resultado do desenvolvimento, ou seja, o produto final, e as possíveis implementações futuras que foram identificadas.

ABSTRACT

This document presents all the development processes of this course conclusion project and its purpose is for Stefanini IT Solutions.

In this document there was an analysis of the client's needs to have a system that would solve some of the company's deficiencies, and automatize existing manual processes related to work flows. There was no control related to the project's documents, consequently the company's products were not well controlled, generating many kinds of risks, such as documentation losses or uncontrolled documentation changes. Nowadays document security is fundamental for a success company.

As the main part, this document presents the methodology that was applied during the development of this project, starting from the client acquisition, project plan elaboration, the complete comprehension of de system requirements, finishing at the subsequent analysis phases, implementation and documentation.

C# (C Sharp) was the development language used to develop the system. As development IDE was utilized the Visual Studio 2005, running under Windows 2000 (or higher) and the SQL Server 2000 database.

It was also presented the result of the development (product) and the future implementation of features and functionalities of the product.

1 INTRODUÇÃO

As recentes exigências legais sobre a gestão de documentos nos mais diversos formatos e a necessidade de eliminar a lentidão nos processos de negócio estão empurrando as empresas para uma nova realidade de integração tecnológica.

Pela junção de diversas ferramentas e métodos, pouco a pouco, o mercado está integrando numa só plataforma o gerenciamento de processos e documentos. Por trás dessas iniciativas estão tecnologias bastante maduras, como o *scanning* e o *workflow*. Juntos, prometem dar mais transparência a informações e processos, elevando o nível de produtividade das corporações.

Projetos de *workflow*, associados às tecnologias de gestão de conteúdos estão ajudando as empresas a serem mais rápidas e gastarem menos papel além de garantir segurança na gestão de registros nas empresas.

Analistas identificam três tipos de projetos neste mercado:

a) os departamentais, com forte impacto financeiro mas restritos a uma área específica;
b) os corporativos, com foco em processos que atravessam várias áreas da empresa;
c) os *enterprises wide*, que integram todos os documentos e processos de uma corporação.

Segundo BAIENSE, "...não se trata apenas de gerenciar arquivos num formato de mídia específico, mas do gerenciamento de todos os registros em todos os tipos de mídia possíveis, desde a criação até sua destruição ou arquivamento permanente."[1] Busca-se otimizar os processos organizacionais.

[1] BAIENSE, C. Tudo num só lugar. **e-Manager**, São Paulo, v. 46, p. 34 - 41, dez. 2003

2 ESPECIFICAÇÃO DO PROJETO

2.1 CARACTERIZAÇÃO DO CLIENTE

Fundada em 1987, e formada pelo grupo de outras 12 empresas, a empresa Stefanini I.T. Solutions, atua em diversos países, fornecendo produtos e serviços em sistemas de informação.

Nestes 20 anos, A Stefanini acumulou várias certificações, tais como a ISO (International Standards Institute) 9001, obtida em 1996, a CMM (Capability Maturity Model), nível II, obtida em 2002, e a CMMI III, obtida em 2004. A empresa obteve ainda vários prêmios, como o título de bicampeã da revista Isto É – Dinheiro como melhor empresa de software e serviços em 2006 e melhor empresa de integração pela revista Info Exame de 2005, em sua pesquisa Info de Marcas.

Vem sendo reconhecida como uma das melhores consultorias estratégicas de Tecnologia da Informação.

2.2 PROBLEMA

As primeiras questões levantadas ao iniciar o planejamento para a realização deste Trabalho de Conclusão do Curso, foram a possibilidade de trabalhar com uma empresa externa à Universidade e se essa opção era aconselhável.

Desenvolver o projeto para uma empresa externa à instituição de ensino ganhou a preferência, pois representava a possibilidade de se fazer um trabalho voltado para o mercado de trabalho, com exigências e prazos reais, e um desafio maior à equipe, fora do ambiente controlado, apresentado pela instituição acadêmica.

Uma vez validada a opção por um cliente externo, deu-se início a pesquisa por possíveis clientes. Um dos membros da equipe verificou na corporação na qual era feito seu estágio que havia a necessidade de um sistema que realizasse a organização de documentos que garantisse agilidade nos processos as quais eles

pertenciam. A proposta foi exposta ao orientador que aceitou que realizássemos esse desafio.

Durante conversas preliminares com representantes da organização, apresentou-se a equipe, que se propôs a desenvolver o sistema para gerenciar os documentos de projetos da empresa, como trabalho para a conclusão de curso para esta equipe, com todas as implicações acadêmicas, idéia a qual foi muito bem aceita pelo possível cliente.

Nas conversas seguintes houve o detalhamento da necessidade da empresa. Apresentou-se o sistema de gerenciamento de projetos da empresa, Dashboard. Esse sistema é a ferramenta utilizada para o controle de todo o processo dos projetos de software realizados pela empresa, como controle de cronograma, recursos e atividades, porém tem um armazenamento falho de documentos. Fora então solicitado que o módulo de carregamentos de documentos do mesmo fosse refeito num sistema integrado, porém separado do original.

No sistema Dashboard, os documentos dos projetos ficam disponíveis para adição, edição e exclusão por todos os integrantes do projeto, sem controle de permissões. Em verdade descobriu-se que esta opção de salvar os documentos no sistema de gerenciamento de projetos era muito pouco usada. De fato, os documentos eram salvos em um servidor munido de software comercial de gerenciamento e versionamento de código-fonte de sistemas computacionais. Sob estas condições, não se prevê que um mesmo documento tenha mais de uma versão salva oficialmente nem que se crie um fluxo de trabalho para o mesmo. Este ambiente é impróprio para o devido controle de documentação, requisito para a empresa alcançar a certificação de qualidade CMMI – Nível 5, objetivo atual da empresa no quesito de certificação. Uma vez levantados estes aspectos, os estudos se ateram em chegar a melhor maneira de implementar os controles solicitados para a efetiva resolução das questões.

2.3 DEFINIÇÃO DO SISTEMA

Avaliando a situação que apresentada pelo cliente, determinou-se que o sistema a ser desenvolvido, o sistema Docboard, precisaria trabalhar basicamente sob dois conceitos: gerenciamento eletrônico de documentos (GED) e Fluxo de Trabalho (*workflow*).

O conceito de GED foi trazido ao Brasil pela empresa Centro Nacional de Desenvolvimento Micrográfico (CENADEM), fundada em 1976, e que ainda hoje tem como um dos principais objetivos difundir tecnologias de gerenciamento de documento. Segundo a CENADEM "...os sistemas de Gerenciamento Eletrônico de Documentos não são simplesmente sistemas de gerenciamento de arquivos. O GED é mais, pois ele implementa categorização de documentos, tabelas de temporalidade, ações de disposição e controla níveis de segurança. É vital para a manutenção das bases de informação e conhecimento das empresas."[2]

Ao sistema GED pensado, agregou-se características de controle de versão, automático para todos os documentos criados. Quando feito o carregamento no sistema as novas versões de documentos, as versões prévias não serão perdidas, mas estarão disponíveis em um histórico, possibilitando inclusive a recuperação de um documento anterior ao que estiver disponível atualmente no sistema, situação na qual o documento atual passaria a ser a penúltima versão e a cópia recuperada de versão anterior assume o status de documento atual. Uma das funcionalidades a serem inclusas no GED para o sistema seria a possibilidade de incluir fluxos de trabalhos (*workflow*) nos documentos criados. De acordo com a definição disponibilizada pela Wikipedia:

> Workflow pode ser definido como seqüência de passos necessários para que se possa atingir a automação de processos de negócio, de acordo com um conjunto de regras definidas. O conceito de Workflow foi concebido de acordo com a noção de processos. Permitindo que estes possam ser transmitidos de uma pessoa para outra de acordo com algumas regras. O gerenciamento de workflow possui um conjunto de ferramentas para administração de monitoramento, para controlar aplicações clientes do workflow, as aplicações invocadas, ferramentas de processos dentre outras. (...). Sistemas de workflow se inserem no contexto geral de software cujo objetivo é o suporte ao trabalho cooperativo, onde se enfatiza a interação entre usuários, e não apenas a interação usuário/sistema.[3]

[2] **O GED.** Disponível em: <http://www.cenadem.com.br/ged01.php> Acesso em 26 mai. 2007.
[3] **Workflow.** Disponível em: <http://pt.wikipedia.com/wiki/Workflow> Acesso em 26 mai. 2007.

Uma vez que estas características foram expostas ao cliente e avaliadas positivamente, iniciou-se então um processo de maior refinamento das funcionalidades, como estarão descritas nos Plano de Gerenciamento de Projetos (PGP), no apêndice 1.

3 METODOLOGIA PARA O DESENVOLVIMENTO DO PROJETO

Pode-se dizer que a metodologia de desenvolvimento abrange todo o ciclo de vida do projeto em questão, incluindo todas as regras criadas, os padrões adotados, ferramentas que serão utilizadas, e todos os pontos relevantes para a definição do processo de desenvolvimento completo.

No projeto Docboard optou-se por utilizar o Rational Unified Process (RUP), por se tratar de uma metodologia com etapas bem definidas e cada uma delas com objetivos bem definidos. Para a RATIONAL SOFTWARE CORPORATION, "...Ele oferece uma abordagem baseada em disciplinas para atribuir tarefas e responsabilidades dentro de uma organização de desenvolvimento. Sua meta é garantir a produção de software de alta qualidade que atenda às necessidades dos usuários dentro de um cronograma e de um orçamento previsíveis."[4]

O RUP não foi aplicado em sua plenitude, por se tratar de uma metodologia bastante extensa e com grande numero de artefatos gerados, leia-se documentos, relatórios ou diagramas. Alguns deles foram considerados desnecessários, fazendo uso somente dos relevantes e fundamentais.

Baseando-se no RUP e filtrando os seus principais aspectos, foi desenvolvido o seguinte modelo, executado como metodologia de desenvolvimento:

a) levantamento de requisitos;
b) análise e design;
c) implementação;
d) teste;
e) implantação.

[4] RATIONAL SOFTWARE CORPORATION. **Rational Unified Process:** Visão Geral. Disponível em: <http://www.wthreex.com/rup/> Acesso em 22 mai. 2007.

3.1 FASES DA METODOLOGIA

3.1.1 Levantamento de Requisitos

Nesta fase foram listados todos os requisitos do sistema, baseando-se na solicitação inicial do cliente. Através desta etapa as funcionalidades necessárias foram listadas e descritas, proporcionando uma base para o desenvolvimento de seu detalhamento e o avanço para as próximas fases.

A elaboração e descrição dos casos de uso ocorre neste momento, as regras de negócio são bem definidas e o protótipo do sistema é elaborado. Neste ponto o cliente avaliou os casos de uso, visando o entendimento de ambas as partes dos objetivos e do escopo do sistema.

Sendo aprovados os casos de uso e o protótipo, a próxima fase pode ser iniciada.

3.1.2 Análise e Design

Com os casos de uso finalizados, iniciou-se a análise do sistema, onde foi definida a sua arquitetura. Nesta definição de arquitetura, foi feita a divisão do desenvolvimento do sistema em camadas, para a melhor divisão e centralização das responsabilidades de cada camada.

Esta divisão foi baseada no modelo em N-Camadas, utilizando na prática o modelo em três camadas. Estas foram especificadas em camada de apresentação ou interface com o usuário, camada de negócio e camada de acesso a dados. A camada de apresentação é a responsável pelas validações de tela e apresentação de dados ao usuário. A de acesso a dados realiza a comunicação direta com os dados armazenados em banco. A de negócio realiza a comunicação entre essas duas camadas, recebendo as requisições da interface, realizando todo o processamento lógico do sistema. Ela entrega e busca informações da camada de acesso a dados e, tratando-as conforme seja necessário, devolve os dados para a camada de aplicação. A camada de aplicação nunca realiza solicitações diretas à camada de acesso a dados, e vice-versa.

Nesta fase também foram desenvolvidos os diagramas essenciais do sistema, como o diagrama de classes, seqüência, de estado, entre outros, e também a modelagem do banco de dados.

3.1.3 Implementação

Possui como objetivo a implementação propriamente dita do sistema modelado, baseando-se na documentação gerada nas fases anteriores. Aqui foram aplicados conhecimentos sobre as regras de negócio especificadas, e feita a modularização do sistema para possibilitar a melhor divisão de tarefas, visando também facilitar a futura integração entre os módulos desenvolvidos.

Foram analisados os casos de uso e os diagramas detalhadamente, e foi implementado exatamente o que estava especificado. Quando da ocorrência de inconsistências na documentação, o assunto foi discutido e, quando necessário, a documentação foi alterada para se adaptar a real necessidade identificada.

3.1.4 Testes

A metodologia foi dividir os testes do sistema entre os membros da equipe, cada um testando o módulo do outro membro e documentando as falhas encontradas.

Antes de colocar o sistema em produção, uma equipe ou um grupo restrito de usuários deve realizar os testes do cliente, visando garantir o correto funcionamento do sistema e também verificar se o produto final foi o que o cliente esperava.

3.1.5 Implantação

Finalmente, o sistema deve ser implantado no cliente, levando em conta o ambiente necessário e os pré-requisitos definidos inicialmente para seu correto funcionamento.

3.2 GERENCIAMENTO DO PROCESSO DE SOFTWARE

"Um processo de software pode ser definido como um conjunto de atividades, métodos, práticas e transformações que as pessoas usam para desenvolver e manter o software e os produtos associados, como planos de projeto, documentos, código, casos de teste e manuais de usuário."[5]

PRESSMAN afirma que "os processos de software devem ter uma forte gerência, através de planos detalhados, estimativas, medições, avaliações e controle."[6]

As atividades técnicas - análise e projeto, algoritmos, estruturação de dados, linguagens etc. - devem ser tratadas com métodos, técnicas e ferramentas específicos, mas a gerência do processo como um todo precisa se utilizar de conceitos e métodos idênticos aos de outras áreas: a definição de objetivos claros para a organização, o comprometimento da alta administração, o envolvimento e o preparo das pessoas, a sincronização de ações em uma mesma direção. As ferramentas para avaliação e controle gerencial também são as mesmas.

O principal objetivo de gerenciar o processo de software é obter qualidade no produto final. Um processo com qualidade resulta em um produto com qualidade.

Segundo PRESSMAN, qualidade de software se define em "Conformidade a requisitos funcionais e de desempenho explicitamente declarados, a padrões de desenvolvimento claramente documentados e a características implícitas que são esperadas de todo software profissionalmente definido."[7]

A gerência do processo visa estabelecer uma infra-estrutura para suportar e guiar os trabalhos dos diversos projetos de maneira uniforme. A gerência inclui:

a) Definição do processo: estabelecer um padrão para implementação, avaliação e melhoria de cada tarefa;
b) Execução do processo: definir os métodos e técnicas usadas para produzir produtos com qualidade;

[5] THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. **IEEE STD 829:** Standard for Software Testing Documentation. New York: IEEE Computer Society, 1998.

[6] PRESSMAN, R. S. **Engenharia de Software**. 6. ed. Rio de Janeiro: McGraw-Hill Interamericana do Brasil, 2006. p. 52.

[7] Ibid p. 65.

c) Coleta de dados e análise: tratar as medições realizadas dos produtos e processos de software, e o uso desses dados;

d) Controle do processo: estabelecer mecanismos para certificar o desempenho do processo definido; monitorar e ajustar o processo onde melhorias forem necessárias.

No projeto Docboard o gerenciamento do processo de desenvolvimento foi definido no PGP, sendo que nele estão detalhados os objetivos, e as ferramentas utilizadas para atingi-los.

Foi feito o uso de métodos auxiliares para a realização das estimativas, como a Análise de Pontos por função, "...é um método padronizado para a medição de projetos de desenvolvimento de software, visando estabelecer uma medida de tamanho, em Pontos de Função (PF), considerando a funcionalidade implementada, sob o ponto de vista do usuário."[8] Utilizou-se também diagramas como o *Work Breakdown Structure* (WBS) para auxílio no desenvolvimento do cronograma do projeto. Foi feito o levantamento de riscos e sua medição de impacto, com o objetivo de minimizar ou extinguir possíveis problemas durante o andamento do projeto.

E para o controle como um todo, foram desenvolvidos relatórios que foram entregues periodicamente, com intenção de controlar as atividades dos componentes da equipe, bem como a definição de datas de entregas de atividades, visando o acompanhamento do cumprimento do cronograma estipulado.

3.3 TÉCNICAS E FERRAMENTAS PARA MODELAGEM FUNCIONAL E NÃO-FUNCIONAL

Primeiramente, deve-se entender o que é um requisito funcional e um requisito não-funcional. Segundo WAZLAWICK, um requisito funcional é o que o sistema deve fazer, e os requisitos não funcionais são as restrições sobre como o sistema deve desempenhar suas funções.[9] Ainda segundo WAZLAWICK, a

[8] HAZAN, C. **Análise de Pontos por Função:** Uma Ferramenta na Implantação do Modelo CMM. Disponível em <http://www.serpro.gov.br/publicacao/tematec/publicacao/tematec/2003/ttec65>. Acessado em 23 mai. 2007.

[9] WAZLAWICK, R. S. **Análise e Projeto de Sistemas de Informação Orientados a Objetos.** 1 ed. São Paulo: Campus, 2004. p.102

modelagem funcional especifica as funções externas do sistema, e descreve as Operações do Sistema como entradas, e as Consultas do Sistema como saídas.[10]

A modelagem funcional do sistema Docboard foi iniciada através do diagrama de casos de uso e de seu detalhamento. Nesse detalhamento estão presentes as pré-condições, o funcionamento, os fluxos alternativos e as exceções.

As pré-condições do sistema definem o que deve existir ou estar presente para que determinada funcionalidade do sistema seja executada corretamente. Os tipos de pré-condições podem ser a existência de parâmetros em sessão para serem utilizados na chamada de uma funcionalidade, ou informações no banco de dados estarem em determinada situação para que outra funcionalidade possa ser ativada.

Visando atender às necessidades da modelagem funcional, foi estipulado o desenvolvimento de diagramas da *Unified Modeling Language* (UML), por possuir diagramas padronizados que facilitam o entendimento do desenvolvedor do sistema, tanto na fase de implantação como para manutenções futuras. Segundo definição de GUEDES:

> A UML (Unified Modeling Language) [sic] é uma Linguagem visual utilizada para modelar sistemas computacionais por meio do paradigma de Orientação a Objetos. Essa linguagem tornou-se, nos últimos anos, a linguagem padrão de modelagem de software adotada internacionalmente pela industria de Engenharia de Software(...)Seu objetivo é auxiliar os engenheiros de software a definir as características do software, tais como seus requisitos, seu comportamento, sua estrutura lógica, a dinâmica de seus processos e até mesmo suas necessidades físicas em relação ao equipamento sobre o qual o sistema deverá ser implantado. Todas essas características são definidas por meio da UML antes do software começar a ser realmente desenvolvido. [11]

Como ferramenta para o desenvolvimento da modelagem funcional, optou-se pelo Rational Software Modeler (RUP), pelo fato de ser uma ferramenta que atende à UML 2.0, e ser uma ferramenta bem estruturada, que possibilita a interação entre os diagramas criados e a melhor organização de todos os diagramas que possam existir no sistema.

[10] WAZLAWICK, loc. cit., p. 102.
[11] GUEDES, G. T. A. **UML:** Uma abordagem pratica. 1 ed. São Paulo:Novatec, 2004. p. 22.

A modelagem não-funcional do sistema foi descrita nos casos de uso, como as regras de negócio, que descrevem restrições e regras propriamente ditas, ou também, por exemplo, quais são os requisitos mínimos para utilizar o sistema, tanto em nível de ambiente, como em nível e conhecimento que os usuários deverão possuir sobre a utilização do mesmo.

3.4 TÉCNICAS E FERRAMENTAS PARA MODELAGEM DE DADOS

A modelagem de dados visa principalmente preparar uma estrutura para a montagem do banco de dados. Nela são definidas as tabelas, chaves primárias, relacionamentos e todos os itens necessários para uma correta representação dos dados.

A ferramenta utilizada para a modelagem de dados foi o Microsoft Visio, por ser de fácil utilização, atender todas as necessidades encontradas e permitir a geração de scripts para a futura criação do banco de dados.

3.5 FERRAMENTAS PARA IMPLEMENTAÇÃO

3.5.1 Requisitos determinados pelo cliente

Conforme consta no PGP, o cliente apresentou alguns requisitos iniciais para a construção do sistema.

Uma das requisições foi que o ambiente de desenvolvimento fosse inteiramente dentro da plataforma Microsoft. O Visual Studio 2005 é a mais nova ferramenta disponibilizada pela Microsoft para a construção de sistemas. Uma das suas maiores características é a utilização de componentes próprios na camada de apresentação e na de acesso a dados, possibilitando uma otimização no desenvolvimento voltado para a orientação a objetos.[12] A IDE tem também diversas ferramentas para análise de códigos e testes.[13]

[12] **What's New in Visual Studio 2005**. Disponível em: <http://msdn2.microsoft.com/en-us/library/88fx1xy0(VS.80).aspx#rtmnewinvs2005> Acesso em 26 mai. 2007.

[13] TARIFA, A. **Qualidade de Código -** Como o Visual Studio 2005 pode nos ajudar! Disponível em: <http://www.linhadecodigo.com.br/artigos.asp?id_ac=1094> Acesso em 26 mai. 2007.

Outra solicitação feita pelo cliente foi a utilização do SQL Server 2000. Atualmente todos os sistemas da empresa estão utilizando esse banco de dados. A empresa optou por não migrar seu banco para uma versão mais recente, pois a segurança de seus dados poderia ser comprometida, já que o release de 2005 ainda não foi amplamente testado. Ela tem intenção de realizar essa migração somente quando mais testes tiverem sido realizados e a versão já tenha mais casos de sucesso.

3.5.2 Métodos de trabalho adotado pela equipe

Buscando atender as requisições do cliente, tornaram-se necessárias as definições de algumas técnicas de trabalho a serem adotadas.

Para utilização do Visual Studio 2005 seria necessária a escolha de uma linguagem que fosse suportada pela ferramenta. Optou-se por utilizar o Visual C#. Suas maiores vantagens são a similaridade com o C++ e todas as características das linguagens orientadas a objetos.[14] Houve também uma indicação do cliente pois, para ele, as linguagens do padrão Visual Basic tendem a ser descontinuadas.

Na utilização do SQL havia 2 métodos que atenderiam a necessidade do projeto. Um deles seria a utilização da sintaxe desenvolvida diretamente na ferramenta, já no outro haveria modularização dos códigos gerados, utilizando-se stored srocedures (SP's). Optou-se pelos SP's ou procedimentos armazenados porque traziam diversas vantagens:

a) Redução do tráfego de rede: "... acontece, porque ao invés da aplicação enviar um grande número de comandos, é enviado para o servidor apenas o pedido de execução do Stored Procedure e os parâmetros de entrada necessários. O SP executa e retorna os resultados, na forma de parâmetros de saída."[15];

[14] **C# Version 3.0 Specification.** Disponível em: <http://msdn2.microsoft.com/en-us/library/ms364047(vs.80).aspx> Acesso em 25 mai. 2007.

[15] BATTISTI, J. **SQL Server 2000:** Administração e Desenvolvimento Curso Completo. 1 ed. Rio de Janeiro: Axcel Books do Brasil Editora Ltda, 2001. p. 569.

b) Melhoria na velocidade de execução: Os comandos são compilados uma única vez e permanecem em memória, o que torna as consultas seqüentes mais rápidas.

Havia também a necessidade decisão do local onde os documentos seriam armazenados. Tinha-se a opção de gravação direta em banco de dados e gravação no sistema de arquivos. Buscando-se otimizar a performance e possibilitar autonomia do administrador do sistema com relação aos backups, optou-se por desenvolver no sistema métodos que salvassem os arquivos no sistema de arquivos. Com isso os arquivos ficariam mais vulneráveis, ficando expostos à possibilidade de vírus e alteração direta através do sistema de arquivos. O controle dos arquivos ficou sob responsabilidade do cliente que fará todos os ajustes necessários para manter a segurança e confiabilidade dos arquivos.

3.6 TÉCNICAS E FERRAMENTAS PARA PROCESSO DE HOMOLOGAÇÃO

3.6.1 Homologação interna

Como método de homologação interna utilizou-se os testes com base em *checklists* pré-definidos pela equipe. Foram realizados testes de forma alternada e em dois ciclos, nos módulos desenvolvidos por outro desenvolvedor. Buscou-se esse procedimento porque aumentaria a possibilidade de falhas serem encontradas no sistema.

3.6.2 Homologação pelo cliente

Essas 2 etapas do desenvolvimento ainda não foram realizadas devido a um atraso do cliente na disponibilização do ambiente de homologação. O processo tornou-se mais complicado devido ao ambiente de teste que será disponibilizado situar-se em São Paulo e o responsável da empresa pelo acompanhamento do processo estar sem disponibilidade.

Houve um agendamento com o cliente para início dos testes até o final do mês de junho deste ano. Tendo ciência da necessidade da entrega do projeto, o

cliente elaborou uma declaração de intenção de uso do sistema, disponível no apêndice 4.

4 PROCESSO DE DESENVOLVIMENTO DA APLICAÇÃO DOCBOARD

4.1 APLICAÇÃO DA METODOLOGIA ADOTADA

A metodologia de desenvolvimento do projeto Docboard foi seguida em todas as suas etapas, porém com alguns desvios durante o processo. A fase de levantamento de requisitos foi realizada sem maiores dificuldades inicialmente, e foi elaborado o protótipo do sistema. Através da análise do cliente deste primeiro modelo, constatou-se algumas divergências de entendimento das regras de negócio, o que ocasionou uma remodelagem de alguns itens dos casos de uso e do protótipo.

Após uma análise mais aprofundada dos casos de uso confeccionados até então, a equipe optou por adotar outro padrão de descrição, adicionando as telas do protótipo no corpo da descrição dos fluxos, visando o melhor entendimento do sistema por parte do cliente.

A diagramação do sistema também sofreu impactos, já que os casos de uso foram alterados. Os diagramas iniciais não representavam a real implementação que seria feita no sistema, já que a equipe optou pelo desenvolvimento em três camadas, sendo elas a camada de apresentação, negócio e acesso a dados. Com isso, os diagramas do sistema também foram modificados, visando representar de forma mais próxima possível a implementação.

Durante a implementação do sistema, chegou-se à conclusão de que alguns itens ainda ficaram pendentes nos casos de uso e nos diagramas, bem como na modelagem dos dados. Esta constatação ocasionou numa nova reformulação de alguns itens, para que a documentação ficasse condizente com o produto final.

Os testes do sistema foram realizados sem dificuldades, utilizando os *checklists* de testes desenvolvidos pela equipe, e após cada ciclo de testes o responsável corrigia as falhas encontradas, até que todos os itens fossem corrigidos.

4.2 APLICAÇÃO DO GERENCIAMENTO DO PROCESSO DE SOFTWARE

Inicialmente foi desenvolvido o PGP com todos os itens e medições necessárias, porém durante o desenvolvimento de projeto o cronograma

estipulado não foi cumprido, visto que alguns requisitos fundamentais do sistema foram alterados.

Ao final da fase de especificação dos diagramas o cliente precisou mudar seu representante no projeto, que ocasionou um atraso nas atividades, por ser necessário um período de adaptação e apresentação do projeto ao novo representante.

A questão que causou um maior impacto no cronograma foi a inclusão do *workflow* no sistema, que inicialmente seria uma ferramenta pronta do cliente integrada ao Docboard para a montagem dos fluxos, e que, após alguns testes, chegou-se à conclusão de que seria completamente inviável utiliza-la, fazendo com que a equipe precisasse desenvolver seu próprio *workflow* que atendesse às necessidades do cliente. O impacto de desenvolver este *workflow* foi extremamente grande, pois com isso o sistema todo teve que ser revisto e remodelado, aumentando em 50% o esforço que da equipe no projeto, impactando diretamente nos prazos. Todos os casos de uso e diagramas auxiliares tiveram que ser modificados, visto que o desenvolvimento de um *workflow* e de funcionalidades de monitoramento das suas atividades não estavam previstos inicialmente.

Um dos integrantes, Marcelo Cabral Matos, deixou de fazer parte da equipe, principalmente devido a questões de integração com os demais membros. Esse imprevisto ocorreu na época em que acreditava-se que o projeto estivesse quase no meio da fase de implementação. Porém, devido a necessidade de ajustes na modelagem para adequação do sistema ao *workflow*, e a complexidade do seu desenvolvimento, verificou-se que somente 20% dessa fase estava finalizada. Os diagramas estavam quase prontos, atendendo ao modelo estipulado pelo grupo, porém a implementação tornou-se mais complexa do que o esperado. As configurações de logon, as funcionalidades de pastas e documentos já estavam quase finalizadas, porém verificou-se a necessidade de ajustes para que se adequassem as reais necessidades encontradas.

4.3 TÉCNICAS E FERRAMENTAS PARA MODELAGEM FUNCIONAL E NÃO-FUNCIONAL

A modelagem inicial do sistema Docboard foi realizada na ferramenta Microsoft Visio, porém, após toda a reformulação do sistema e o aumento de experiência dos membros na equipe, optou-se por mudar a ferramenta de modelagem para o Rational Software Modeler (RSM), por atender amplamente às necessidades da equipe e atender à padrões da UML 2.0.

A seqüência temporal do desenvolvimento da modelagem iniciou-se com os casos de uso, com o levantamento dos requisitos e o desenvolvimento do protótipo no Visual Studio 2005, mesma ferramenta utilizada na implementação do sistema. Os documentos gerados estão no apêndice 5, que contem os Casos de uso, com as telas desenvolvidas no protótipo.

Para atender as necessidades da modelagem funcional foi desenvolvido também o diagrama de classes, que é o diagrama mais utilizado e o mais importante da UML, servindo de apoio para a maioria dos outros diagramas. [16]

Em seguida, foram desenvolvidos os diagramas de seqüência do sistema, baseando-se nos casos de uso e nos diagramas de classes. Segundo GUEDES, o diagrama de seqüência preocupa-se com a ordem temporal em que as mensagens são trocadas entre os objetos envolvidos em um determinado processo. Em geral, baseia-se em um caso de uso definido pelo diagrama de mesmo nome e apóia-se no diagrama de classes para determinar os objetos das classes envolvidas em um processo.[17]

Como diagramas de apoio, utilizou-se dos diagramas de Máquina de Estados, que "... procura acompanhar as mudanças sofridas por um objeto dentro de um determinado processo."[18], e dos diagrama de Componentes, que "... representa os componentes do sistema quando este for ser implementado em termos de módulos de código fonte, bibliotecas, formulários, módulos executáveis, etc. e determina como estes componentes estarão estruturados e interagirão para que o sistema funcione de maneira adequada."[19]

Os diagramas confeccionados durante esta fase podem ser verificados no apêndice 6, que apresenta o diagrama de classes; apêndice 7, que contem os

[16] GUEDES, op. cit., p. 27.
[17] GUEDES, loc. cit., p. 28.
[18] Id.
[19] Ibid., p. 30.

diagramas de Seqüência; apêndice 8, que contem os diagramas de Estado e no apêndice 9 que contem o de Componentes.

4.4 APLICAÇÃO DAS TÉCNICAS E FERRAMENTAS DE MODELAGEM DE DADOS

Utilizou-se a ferramenta definida na formulação do projeto. Verificou-se logo no início da modelagem do Diagrama de Entidades e Relacionamentos (DER) que não seria possível a utilização de integridade referencial com os dados do banco de dados do sistema Dashboard, devido à restrição do SQL Server em aplicar essa técnica em bancos diferentes. Optou-se por garantir essa integridade somente através da aplicação.

Buscou-se realizar a modelagem completa antes do início do desenvolvimento, para o aproveitamento do recurso da ferramenta de geração utomática dos scripts do banco. Porém durante o desenvolvimento boa parte da estrutura de campos e mesmo de tabelas foi alterada, de acordo com as necessidades que foram verificadas, o que fez com que após o término da construção do sistema fosse gerada uma nova versão do DER.

Como artefatos produzidos nessa fase do projeto tem-se o DER, disponível no apêndice 10, e o Dicionário de dados, que segue no apêndice 11.

4.5 APLICAÇÃO DAS FERRAMENTAS DE IMPLEMENTAÇÃO

A ferramentas definidas na formulação do projeto para implementação do sistema foram utilizadas, buscando otimizar a reutilização de classes, conforme a técnica de orientação a objetos indica. Houve, porém, algumas situações que causaram atraso no desenvolvimento.

Devido a falta de experiência da equipe em trabalhar em projetos houve a necessidade de re-trabalho das atividades de implementação. Foram detectadas 2 situações que causaram esse problema:

a) Falta de um diagrama na modelagem do sistema que realizasse um detalhamento maior das regras de como o sistema deveria ser

desenvolvido pelo programador, isto é, detalhamento da lógica das funcionalidades a serem desenvolvidas;

b) Alguns métodos tinham dependência da criação de outros, porém não se conseguiu identificar essa necessidade antes do desenvolvimento.

Buscou-se realizar todos os diagramas da UML tidos como importantes para a modelagem do sistema, porém nenhum deles atendeu a necessidade de um detalhamento maior nas regras a serem utilizadas. A definição da lógica ficou a cargo do programador durante a criação, sem ter havido qualquer definição pré-estabelecida. Apesar de toda a análise ter sido realizada pelos programadores, essa pré-definição teria sido útil para agilizar o processo. Acabou não sendo realizada, pois só verificou-se sua necessidade ao final da implementação.

O método de trabalho por conexão remota mostrou-se muito vantajoso por centralizar as alterações e possibilitar que, sempre que necessário, o programador resgatasse a versão mais recente da classe desejada. Porém houve dificuldade com relação à centralização do banco de dados, pois não foi possível disponibilizar a conexão remota desse serviço. Isso fazia com que muitas vezes SP's desatualizadas e a necessidade de criação ou alteração de campos atrapalhassem na realização do desenvolvimento.

O desenvolvimento do *workflow* do sistema foi realizado em integração com as demais funcionalidades, tendo seus dados armazenados em banco. Buscou-se gerar uma ferramenta específica, que atendesse diretamente à necessidade da aplicação. Foi necessária também a geração de uma *Engine* para monitoramento das atividades do *workflow* e do sistema, que foi desenvolvida usando-se a tecnologia dos Serviços do Windows. Essa foi a opção selecionada devido a ser uma aplicação leve, que exige menor processamento e pode ser realizada diversas vezes por dia, de acordo com os parâmetros pré-estabelecidos no sistema.

A ajuda do sistema foi desenvolvida em HTML, tendo como base os diagramas de caso de uso. Buscou-se organizá-la na seqüência em que as atividades devem ser realizadas no sistema, separando-se as atividades administrativas das operações normais do sistema.

A documentação do sistema não pode ser gerada a partir da ferramenta Visual Studio 2005, como foi definido. Isso aconteceu porque a opção de geração

automática foi retirada dessa versão, estando disponível somente na versão 2003 da IDE. Foi utilizada a ferramenta Doxygen versão 1.5.2, produzido pela empresa de mesmo nome, para esse fim, pois, apesar de ser voltada para a documentação de códigos em C++ utiliza-se das mesmas tags de comentário que são sugeridas para C#.[20]

Os artefatos gerados nessa fase são:

a) código-fonte do sistema, contendo as classes de desenvolvimento e a ajuda incorporada no site. Está disponível no apêndice 12;
b) engine do sistema, disponível no apêndice 13;
c) script do banco de dados, contendo métodos de criação de todas as tabelas e SP's do sistema, que está no apêndice 14;
d) documentação do sistema, no apêndice 15.

4.6 APLICAÇÃO DAS TÉCNICAS DE HOMOLOGAÇÃO INTERNA

Durante toda a etapa de desenvolvimento, cada programador realizou os testes iniciais no módulo desenvolvido (teste unitário). Após o término do processo, foram preparados os *checklists* de teste por funcionalidade. A elaboração dos *checklists* foi realizada buscando atender os seguintes requisitos:

a) os parâmetros estipulados nos casos de uso deveriam ser observados;
b) as configurações de tela buscavam corresponder às realizadas no protótipo;
c) verificações nos dados alterados no banco e em diretórios físicos deveriam ser feitas.

Foram realizados dois ciclos de teste interno com base nos *checklists*, alternando-se os responsáveis pelos testes entre os membros da equipe, garantindo-se que o desenvolvedor não realizasse os testes no código-fonte gerado por ele. Os ajustes e um último ciclo de testes foram realizados, tendo todos os itens sido atendidos conforme esperado.

[20] BATTISTI, J. **Criando documentação usando xml e comentários.** Disponível em: <http://www.juliobattisti.com.br/tutoriais/herbertgois/programandocsharp004.asp> Acesso em 25 mai. 2007.

Como produto dessa fase foram realizados 3 *checklists* de testes por funcionalidade do sistema, que seguem no apêndice 16.

5 HOMOLOGAÇÃO PELO CLIENTE

Para a homologação pelo cliente ser realizada será necessária a disponibilização do sistema em um servidor de testes, o que possibilitará a verificação de alguma incompatibilidade do sistema no ambiente do cliente.

Serão solicitados os testes ao cliente, que deverá informar as atividades testadas e os resultados conseguidos no formulário de teste contido no apêndice 1.

Após o primeiro ciclo de testes o sistema sofrerá os ajustes solicitados pelo cliente, para que possa ser submetido a um segundo ciclo. A estimativa é de que sejam necessários no máximo três ciclos, mas serão realizados quantos forem necessários até ter-se uma resposta positiva.

6 ESTRATÉGIAS DE IMPLANTAÇÃO

Um requisito para implantação do sistema será a aplicação dos passos do documento de requisitos de implantação do sistema, no apêndice 17. Indica-se que o servidor da aplicação, para disponibilização das páginas, seja separado do servidor de banco de dados, por questões de desempenho.

Devido à necessidade de integração entre os sistemas Docboard e Dashboard, será necessário que o sistema Dashboard disponibilize alguns dados do usuário por POST. Fica a cargo da empresa cliente realizar alterações no sistema Dashboard para que possa ser realizado o login no sistema Docboard automaticamente. Os dados necessários para realização do POST estão contidos no apêndice 17.

7 IMPLEMENTAÇÕES FUTURAS

Durante o planejamento e a execução do projeto Docboard, surgiram algumas possibilidades de novas funcionalidades ou uma nova forma de implementar as funcionalidades já estipuladas.

Como o planejamento já havia sido realizado, as funcionalidades e suas implementações todas definidas e aprovadas para o cliente, não houve a possibilidade de implementar tais funcionalidades. Por este motivo, planejou-se que, havendo uma nova versão, estas funções seriam avaliadas pelo cliente para inclusão ou modificação.

Entre as possibilidades de implementações futuras, encontram-se:

a) Exportação e importação de definições de *workflows* de/para arquivos XML;
b) Edição de templates já cadastrados no sistema;
c) Edição de documentos que não estejam em fluxo;
d) Visualizacao de documentos pelo navegador de internet, em formato HTML, ao invés de abrir o arquivo em seu formato original;
e) Visualização das diferenças entre os arquivos versionados;
f) Ao criar uma pasta, dar acesso ao menos a quem esta criando, para facilitar a posterior edição de permissões;
g) Estudar a viabilidade de se criar uma estrutura de pastas na máquina do usuário, para que o usuário não precise escolher o diretório que deseja realizar o *check-out*, e o *check-in* seja automático, sem o usuário precisar buscar o documento;
h) Compactação dos arquivos físicos para otimizar o espaço disponibilizado pelo cliente;
i) Listar os templates nas telas de listagem de documentos das pastas;
j) Desenhar o fluxo em seu status atual utilizando Scalable Vector Graphics (SVG) ou outra tecnologia que adeque-se bem a essa funcionalidade.

CONCLUSÃO

O sistema de Gerenciamento Eletrônico de Documentos colabora para o bom andamento dos trabalhos de uma organização ao agilizar todo o fluxo que um documento deve percorrer e garantir a integridade do mesmo.

A tarefa de modelar um GED, escolhendo suas funcionalidades desejadas, é essencial para o sucesso do empreendimento.

Quando bem elaborado e devidamente utilizado, um GED torna a empresa mais competitiva e amplia o tempo disponível para as tarefas mais importantes da organização, aumentando consideravelmente o nível de qualidade final dos produtos gerados.

A inserção da equipe em todo este processo trouxe aos membros uma nova visão do mercado de desenvolvimento de software e a experiência muito enriquecedora do trabalho em equipe, do dia-a-dia de um projeto, além de introduzir métodos de medir riscos e evitar falhas, o que muito agrega ao conhecimento de todos, agora melhor preparados para o mercado de trabalho.

GLOSSÁRIO

Administrador: Responsável pela manutenção de permissões no sistema.

Artefatos: São os produtos finais ou intermediários produzidos e usados durante o projeto.

Casos de uso: Diagrama utilizado na informática para representação do sistema a ser desenvolvido, com intenção de facilitar o entendimento do sistema tanto pela equipe de desenvolvimento quanto pelo cliente.

Check-In: Liberação do documento após edição.

Check-Out: Solicitação de permissão para edição do arquivo. Quando concedida o documento só pode ser alterado pelo solicitante até que o documento seja liberado.

Engine: Sistema que tem por objetivo monitorar informações com intuito de impedir que processos fiquem parados.

Metodologia: Maneira adotada para nortear uma tarefa com base em princípios já estabelecidos.

POST: Método disponibilizado no HTML que possibilita a transmissão direta de dados para uma página que esteja preparada para receber essas informações.

Rollback: Possibilidade de recuperar documento da versão anterior e torná-lo ativo.

Sistema de arquivos: Comumente utilizado para representar o disco rígido dos computadores. Nesse documento indica que os arquivos serão armazenados diretamente em pastas no disco rígido, ao invés de serem guardados no banco de dados.

Status do Workflow: Informações sobre o ciclo do documento no projeto. Se aguardando liberação, disponível, etc.

Template: Modelo de arquivo em formato de extensão Office que não será alterado pelo sistema Docboard.

Workflow: É um sistema onde uma atividade pode ser transferida de uma pessoa para outra através de um sistema de rede. Deve atender a um conjunto de regras e tem a função de agilizar os processos na corporação.

REFERÊNCIAS

1 BAIENSE, C. Tudo num só lugar. **e-Manager,** São Paulo, v. 46, p. 34 - 41, dez. 2003.

2 **O GED.** Disponível em: <http://www.cenadem.com.br/ged01.php> Acesso em 26 mai. 2007.

3 **Workflow.** Disponível em: <http://pt.wikipedia.com/wiki/Workflow> Acesso em 26 mai. 2007.

4 RATIONAL SOFTWARE CORPORATION. **Rational Unified Process:** Visão Geral. Disponível em: <http://www.wthreex.com/rup/> Acesso em 22 mai. 2007.

5 THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. **IEEE STD 829:** Standard for Software Testing Documentation. New York: IEEE Computer Society, 1998.

6 PRESSMAN, R. S. **Engenharia de Software**. 6. ed. Rio de Janeiro: McGraw-Hill Interamericana do Brasil, 2006. p. 52.

7 HAZAN, C. **Análise de Pontos por Função:** Uma Ferramenta na Implantação do Modelo CMM. Disponível em: <http://www.serpro.gov.br/publicacao/tematec/publicacao/tematec/2003/ttec65> Acesso em 23 mai. 2007.

8 WAZLAWICK, R. S. **Análise e Projeto de Sistemas de Informação Orientados a Objetos.** 1 ed. São Paulo: Campus, 2004. p. 102.

9 GUEDES, G. T. A. **UML:** Uma abordagem pratica. 1 ed. São Paulo:Novatec, 2004. p. 22.

10 **What's New in Visual Studio 2005**. Disponível em: <http://msdn2.microsoft.com/en-us/library/88fx1xy0(VS.80).aspx#rtmnewinvs2005> Acesso em 26 mai. 2007.

11 TARIFA, A. **Qualidade de Código -** Como o Visual Studio 2005 pode nos ajudar! Disponível em: <http://www.linhadecodigo.com.br/artigos.asp?id_ac=1094> Acesso em 26 mai. 2007.

12 **C# Version 3.0 Specification.** Disponível em: <http://msdn2.microsoft.com/en-us/library/ms364047(vs.80).aspx> Acesso em 25 mai. 2007.

13 BATTISTI, J. **SQL Server 2000:** Administração e Desenvolvimento Curso Completo. 1 ed. Rio de Janeiro: Axcel Books do Brasil Editora Ltda, 2001. p. 569.

14 BATTISTI, J. **Criando documentação usando xml e comentários.** Disponível em:
<http://www.juliobattisti.com.br/tutoriais/herbertgois/programandocsharp004.asp>
Acesso em 25 mai. 2007.

APÊNDICE 1 – PLANO DE GERENCIAMENTO DE PROJETOS

APÊNDICE 2 – ADENDO AO PLANO DE GERENCIAMENTO DO PROJETO

APÊNDICE 3 – RELATÓRIOS INTERNOS MENSAIS

APÊNDICE 4 – DECLARAÇÃO DE INTENÇÃO DE UTILIZAÇÃO

APÊNDICE 5 – DIAGRAMAS DE CASOS DE USO

APÊNDICE 6 – DIAGRAMA DE CLASSES

APÊNDICE 7 – DIAGRAMAS DE SEQUÊNCIA

Disponível no CD que integra o presente documento

Pasta Diagramas

APÊNDICE 8 – DIAGRAMA DE ESTADO

APÊNDICE 9 – DIAGRAMA DE COMPONENTES

APÊNDICE 10 – DIAGRAMAS DE ENTIDADE-RELACIONAMENTO

APÊNDICE 11 – DICIONÁRIO DE DADOS

APÊNDICE 12 – CÓDIGOS-FONTE DO APLICATIVO WEB

Disponível no CD que integra o presente documento

Pasta Códigos-Fonte

APÊNDICE 13 – CÓDIGOS-FONTE E EXECUTÁVEL DA ENGINE

Disponível no CD que integra o presente documento

Pasta Códigos-Fonte

APÊNDICE 14 – SCRIPTS DO BANCO

Disponível no CD que integra o presente documento

Pasta Scripts do Banco

APÊNDICE 15 – DOCUMENTAÇÃO DO CÓDIGO-FONTE

Disponível no CD que integra o presente documento

Pasta Documentação

APÊNDICE 16 – CHECKLISTS DE TESTE INTERNO

APÊNDICE 17 – DOCUMENTO DE IMPLANTAÇÃO